2013

LE PENDU : Client-Serveur
Détails d'implémentation et d'utilisation du service et de l'application

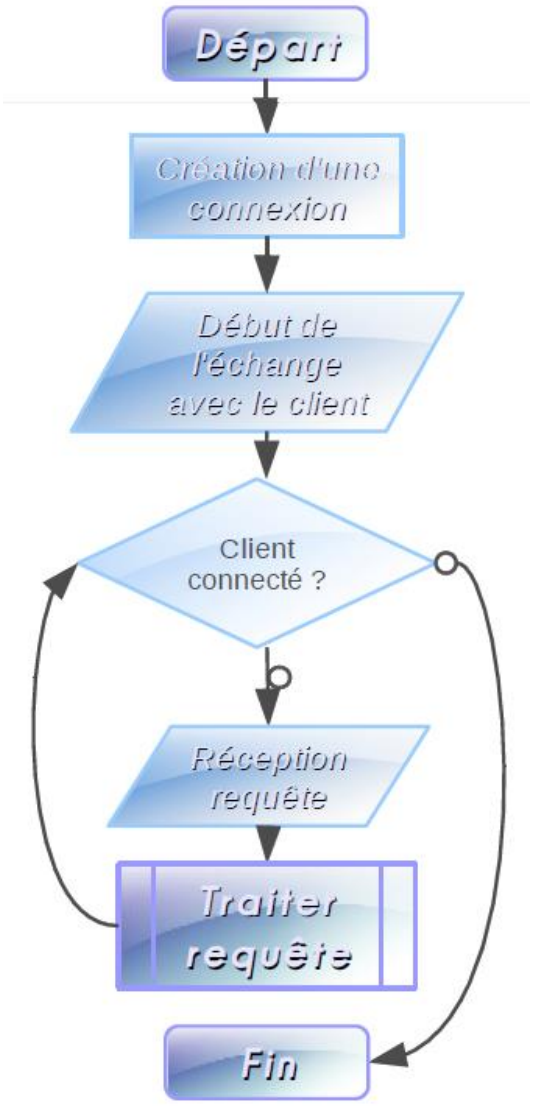
ZIZAH MOHAMMED

DETAILS D'IMPLEMENTATION ET D'UTILISATION DU SERVICE ET DE L'APPLICATION « LE PENDU «

Introduction

Principe du jeu du pendu : un joueur J détient un mot et l'autre joueur doit le deviner. Pour ce faire, il soumet des lettres (une par une) à J qui place la lettre dans le mot si possible et leur indique le résultat de ce placement.Le joueur dispose d'un maximum M de coups à jouer. (Dans notre application M=10)

Vue générale



II. Implémentation

Le service et l'application ont été codé en JAVA, les exécutables sont en .jar.

Le service :

Le service dispose d'une fenêtre (servant à afficher le nombre de connexions au serveur).

Il dispose de 4 classes :

- **Authentification** : connexion du client :si le serveur est complet (100 joueurs) le joueur est déconnecté. Enregistrement du pseudo et du rôle souhaité (gestion des erreurs, tant que le pseudo ou le rôle n'est pas accepté, on redemande).

- **Client** : répertorie les clients qui se connectent avec succès (socket, pseudo, rôle, etc...) et permet d'implémenter le ChoixAdversaire.

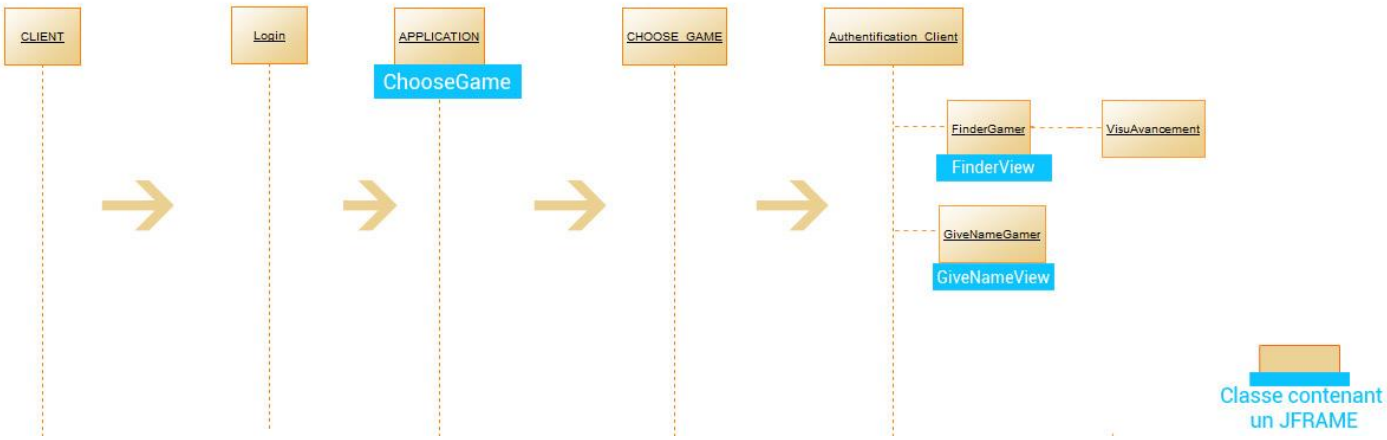
- **ChoixAdversaire** : après la connexion, le joueur passe en « phase d'attente ». Selon son rôle, il va rechercher en boucle un adversaire avec le rôle complémentaire.

- **Pendu** : c'est l'implémentation du jeu en lui-même, où l'on va attendre qu'un mot soit choisi par le joueur qui a le rôle « Choisir mot », puis recevoir les propositions et envoyer les réponses au joueur qui a le rôle « Deviner mot » (une copie de ce qui est rçu et envoyé au joueur « Deviner mot » est retransmie au joueur « Choisir mot » afin qu'il sache comment se déroule la partie).

L'application :

1. Aspect général

Le schéma ci-dessous présente l'appel des classes dans un ordre chronologique correspondant au déroulement du programme.



La classe Client (Main) :

- ✓ Stocker les attributs de connexion
- ✓ Lancer la Fenêtre de connexion : Login
- ✓ Vérifier le login
  a. S'il ne contient pas un ' ;'
  b. S'il n'est pas vide
- ✓ Si le login est bon, lancer la classe Application.

La classe Application

- ✓ Lancer la Fenêtre ChooseGame
- ✓ Choisir un jeu
- ✓ Lancer AuthentificationClient (pour choisir un rôle)
- ✓ Lancer la fenêtre du rôle choisi

2. Aspect technique

Lexique important

Les types de variables les plus importantes :

- **Les sockets** servent à communiquer entre deux hôtes (Client, Serveur) à l'aide d'une adresse IP et d'un port .Ces sockets permettront de gérer des flux entrant et sortant afin d'assurer une communication entre les deux (le client et le serveur)

• **BufferedReader** : cette classe permet de lire des caractères à partir d'un flux tamponné, afin de faire des lectures plus rapides ;

• **InputStreamReader** : convertit un flux binaire en flux de caractères : elle convertit un objet de type InputStream en objet de type Reader ;

• **PrintWriter** : la classe PrintWriter ajoute à un flux la possibilité de faire des écritures sous forme de texte des types primitifs Java, et des chaînes de caractères.

---

1. Tout d'abord dans la classe main on définit l'adresse du host et le port
2. On crée un Socket (new Socket (host,port))

3. Une fois que le login est validé on lance le thread principal (Application)
   - t1 = new Thread(new Application(socket, login)
     (La classe application prend en paramètre le socket créé au départ)

   - Grace au PrintWriter on envoi le login tapé au serveur via la fonction write puis il faut faire un flush (indispensable !)qui force à lancer les écritures et vider le buffer.

   - Le serveur reçoit à son tour le login grâce au BufferedReader par la fonction readLine (une instruction bloquante, c'est à dire tant qu'elle ne reçoit pas de valeur l'instruction qui la suit ne sera pas exécuté)
     Il existe notamment d'autres fonctions de lecture comme readInt … mais j'ai préféré d'utiliser readLine qui fonctionne pour tous types de data qu'on reçoit

4. Apres avoir choisis un jeu parmi la liste on ouvre un autre thread : Authentification Client qui permet de choisir un rôle (Proposer un mot ou deviner le mot déjà proposé).
   Une fois le thread exécuté (dans le run) on initialise le BufferedReader, l'inputStream et le printWriter

5. Dès que l'utilisateur choisi un rôle je fais un switch case pour déterminer le rôle à attribuer, puis je crée un thread vers l'interface adéquate

   i. On initialize PrintWriter et BufferedReader
   ii. Si la connexion avec le serveur est établie avec succès, on attend que l'adversaire soit connecté via la commande bloquante BufferedReader.

readLine qui doit nous retourner un message (qui varie selon le Protocol) qui nous informe que l'adversaire est bien connecté.

iii. L'utilisateur propose un mot puis valide son choix.

6. La dernière partie du programme est la plus intéressante à analyser
   Effectivement à ce niveau-là, le joueur sera amené à proposer une lettre à la fois puis attendre le retour du serveur.

   La technique utilisée est la suivante. On lance un thread dans le coté du joueur un et du joueur 2 pour assurer la communication des deux parties avec le serveur.
   Les deux threads utilisent des fonctions bloquantes (readLine) afin de recevoir la lettre puis l'état de la proposition. On arrête les threads à la fin du jeu c'est à dire soit si le joueur a pu deviner l'intégralité des lettres soit il a épuisé ses vies.

   En effet le but c'est de communiquer aux deux joueurs l'évolution du jeu en temps réel, c'est-à-dire leurs afficher selon l'état de la lettre proposé le nouveau squelette du mot à deviner.

   Les différents états sont :

   - Proposition invalide
   - Proposition juste
   - Proposition fausse

   i. Selon les critères du serveur si la proposition est invalide, l'utilisateur sera amené à ressaisir une nouvelle lettre.
   Cette fonctionnalité est utile dans le cas où on implémente le jeu en console, en effet l'utilisateur pourra saisir n'importe quoi .Une implémentation graphique nous garantit que les lettres saisis par l'utilisateur soient valident

   ii. Si la proposition est juste, le serveur renvoi l'emplacement de la lettre proposé dans le mot a deviné.
   J'ai créé un char* « proposition » de longueur égale à celle du mot à deviner et initialisé avec des '_' .
   Ce tableau est rempli au fur et à mesure que l'utilisateur donne des propositions exactes. Ce tableau m'a été pratique pour l'affichage des résultats en temps réel (par exemple si le mot à deviner était « exemple » et l'utilisateur saisi la lettre 'e' , « proposition » ressemblera à « e_e_ _ _ e »

   i. Si l'utilisateur atteint ses points d'erreurs (10) , il retournera vers la fenêtre « Application » pour choisir un jeu , tout en gardant son login de départ

III. Le manuel d'utilisation de l'application

Le service :

Côté code :

Pour utiliser le service, il faudra utiliser le port 55000 lors de la connexion de la socket et respecter le protocole donné en annexe.

Côté jeu : il faudra d'abord démarrer le service, puis l'application. La partie s'achève lorsque le joueur a deviné le mot adverse ou lorsque les 10 points d'erreurs sont épuisées.

L'application :

Connexion :

L'utilisateur saisi son login puis valide en cliquant sur « connexion » :Figure 2.0

Figure 2.0

Choix du jeu :

L'utilisateur est amené à choisir un jeu parmi ceux proposé ci-dessous
Pour l'instant, le jeu du pendu est le seul service qui fonctionne : Figure 2.1

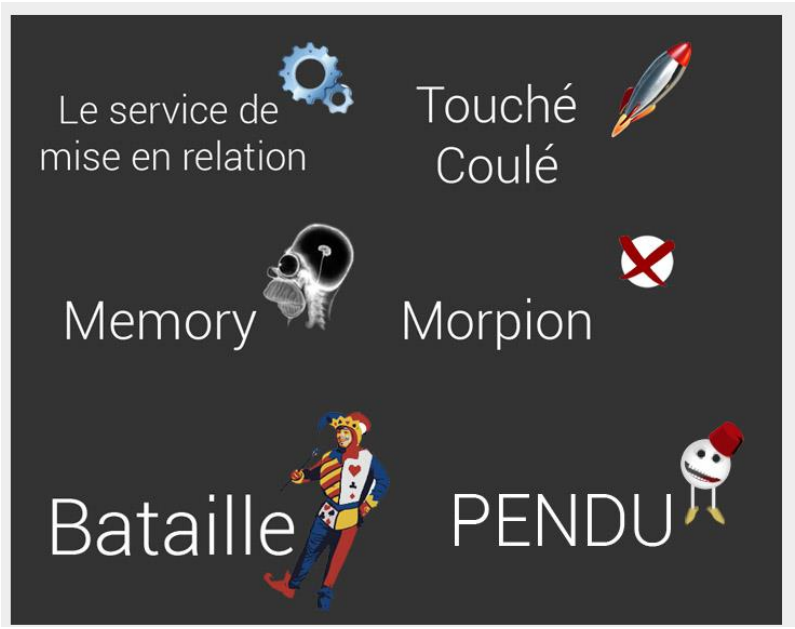


Figure 2.1

Choix de rôle

Une fois le jeu est choisi, l'utilisateur est amené a déterminé son rôle Figure 2.2

Figure 2.2

Rôle : Maitre

Le maitre est la personne qui propose le mot que l'adversaire aura à cherché plus tard

L'utilisateur valide sa proposition de mot :

Si la proposition est valide (Au niveau du serveur), le jeu continue normalement c'est à dire il donnera la possibilité à son adversaire de proposer des lettres à son tour.

Sinon l'utilisateur sera amené à ressaisir un autre mot. Figure 2. 3

Figure 2.3

Rôle : Soumis

Le soumis est la personne qui devine le mot proposé par l'adversaire

Les « _ » correspondent au squelette du mot a deviné.

La barre orange situé à droite de la fenêtre, correspond au nombre de « vies » restantes au joueur

Cœur du jeu

C'est la partie la plus importante du programme, effectivement l'échange le plus intéressant est fait à ce niveau-là qui se déroule.

1. Le client 1 reçoit le squelette du mot a deviner
2. Il propose une lettre depuis le clavier visuel
3. Si la proposition est juste, la lettre sera grisée de manière qu'il ne puisse ressaisir la même lettre une seconde fois .Le squelette ainsi que la jauge de vie se mettront à jour sur la fenêtre des deux clients à la fois.
4. Le jeu prend fin dans deux cas précis, soit quand le client 1 épuise son nombre de vies ou bien quand il trouve le mot en entier.
   Dans les deux cas chaque client reçoit un message annonçant sa victoire ou sa défaite